Eco-Logic: Logic-Based Approaches to Ecological Modelling

Books

Daniel L. Schmoldt

Eco-Logic: Logic-Based Approaches to Ecological Modelling

D. Robertson, A. Bundy, R. Muetzelfeldt, M. Haggith, and M. Uschold, 1991
MIT Press, Cambridge, Massachusetts
243 pages, $35.00
ISBN 0262181436

In the preface to the this book, the authors point out that simulation models can be useful tools for investigating real phenomena in an idealized way. However, the complexity of simulation models can often deter their use by ecologists. Also, complexity and implementation details of a simulation may obscure assumptions underlying the model. These observations provide partial justification for research carried out during 1984-1989 at the University of Edinburgh. This book contains a summary of that research. Two primary objectives for that work are 1) to provide tools for manipulating simulation models (i.e., implementation tools) and 2) to provide advice on conceptualizing real-world phenomena into an idealized representation for simulation (i.e., model design).

Chapter 1 discusses modeling in very general terms. The authors specify three standards that they feel all simulation models should satisfy. These standards are based on the idea that a simulation model is essentially an argument proposed by the modeler about how some real-world system operates. They then show why conventional simulation models cannot meet those standards (i.e., those models are unable to explicitly characterize such an argument), especially in poorly understood subject areas. By "conventional," the authors mean the direct implementation of a model in one of the imperative programming languages, such as FORTRAN, Pascal, or C. Alternatively, because a simulation model consists of assumptions (axioms) and rational arguments (inference methods), the authors reason that logic and logic programming can provide an excellent mechanism for representing and reasoning about simulation models.

The primary focus of this book is on a logic-based approach to simulation modeling. Ecology was chosen as the experimental arena in which to apply these modeling ideas. In Chapter 2, the authors briefly introduce ecology and discuss some of the attributes that make it a complex and poorly understood science. They also present uses of ecological models, types of ecological models, and major modeling paradigms. Any new formalism for ecological modeling will have to take into account the diversity of these three areas.

Prolog is introduced in Chapter 3 as one of the possible logic programming languages for the purposes of simulation modeling. The authors maintain that model comprehension is improved by logic because of its explicit structure, modularity, and flexibility of use. Three simple simulation models are constructed in Prolog to illustrate these points: 1) a system dynamics model, 2) a structural growth model, and 3) a state transition model. Also, the authors translate a conventional simulation model into Prolog. Without some prior experience with Prolog, it will be difficult for the reader to follow and understand these examples. The authors acknowledge this, and refer the reader to several introductory texts

on Prolog for the necessary background.

The authors note two problems that surface with the use of native Prolog as a simulation modeling language. First, because Prolog is a relational language, it will find all solutions for a particular goal unless it is explicitly "told" otherwise (the "cut" operator performs this function). This means that non-declarative information, in the form of cuts, must be included in clauses, and, consequently, the logical, argument-based interpretation of the simulation model specification becomes obscured. Second, when Prolog instantiates variables, it has unrestricted use of all terms in the Prolog database. Unless a programmer exercises caution by carefully ordering clauses and inserting judicious cuts, variables can be instantiated to objects (terms) that may make no sense for a particular clause. The remainder of Chapter 4 presents an order-sorted logic for eliminating these two problems and providing a clean and declarative model representation language.

In essence, a *sort* is a subset of the universe of discourse. In Prolog, the universe of discourse for each variable instantiation is the set of all terms. Prolog and first-order predicate logic are therefore both one-sorted logics. In many-sorted logics, predicates and functions are defined to apply only to particular sorts, or subsets of the terms available. When some ordering of the sorts is provided, it is referred to as an order-sorted logic.

The authors define a sort hierarchy of objects for simulation models. Their order-sorted logic also contains function and predicate declarations and sorted-logic axioms. Using this presentation, any simulation model can now be expressed as a sort hierarchy along with a set of axioms describing the relationships between objects in that hierarchy. A sorted-logic interpreter is provided by the authors in pseudo-code form. A sorted-logic-to-Prolog translator is also described in pseudo code. By including constraints in each axiom, the order of model statements is no longer important (therefore procedural code can be eliminated). Also, functions are now applied only to objects for which they have meaning.

Chapter 5 represents a diversion from the main topics of the book. It delves into considerations of execution efficiency for their order-sorted logic implementation. In their initial implementation, values of time-dependent variables can only be calculated in terms of the variables' values at previous time points. This forces long recursive evaluations and greatly slows down execution speed, because previously calculated values are not retained. Two solutions are explored here: 1) a meta-interpreter for Prolog that stores the results of function evaluation and 2) translation of the sorted logic program into a procedural programming language. The authors highlight several difficulties concerning translation of a declarative language into a procedural one, so they prefer the former solution for their particular sorted logic.

Chapters 6 and 7 address the second objective for this book, namely, to provide a mechanism for non-programmers to transform conceptual models into simulation models. This is a much more difficult problem than the first objective tackled in Chapters 3 and 4. Here, the authors must tread a fine line between a simulation design/generator that is too general to be of use to a non-programmer and one that is too specific to be of use in more than a very restricted subject area.

The sorted logic axioms introduced in Chapter 4 are used in Chapter 6 as feature descriptors for a problem description of any ecological system. This problem description level constitutes a high-level representation that falls somewhere between the implementation level of a simulation and the ecologist's abstract mental model of the real-world system. Deduction, abduction, and consistence y checking algorithms help to automate the problem description process.

Chapter 7 presents the Eco-Logic (EL) system as an intelligent front end to the model tools developed in the previous chapters. EL consists of two subsystems: a problem description system that helps an

EL user construct ecological simulation models in a high-level description language, and a program generation system that produces Prolog code from that description. EL begins with an initial sort hierarchy, ecological and modeling rules, and some input templates. A user may include additional sorts and objects, and can easily edit input templates to create sorted logic axioms. EL completes the user's problem description by applying modeling rules and consistency checks. EL then uses Prolog-program schemata to construct a program from the problem description. An extensive example is presented to illustrate how this process works. Without it, the reader would be completely lost. The program construction system creates three files, one of which can be run by a standard Prolog interpreter. An ancillary reconstruction system can reproduce an entire EL session from the output in the other two files. The authors briefly describe two other systems, HIPPIE and NIPPIE, as enhancements to the EL system.

Recognizing that the process consisting of problem description and program generation does not guarantee that a final program will be executable, the authors decide to modify their system to use a dynamic set of axioms rather than the static set created in the problem description. Then, if the simulation goal contained in the model cannot be satisfied when these axioms are executed by the sorted logic interpreter, the user can insert additional axioms until the goal is solved. At this point the model is necessarily complete.

This alternative modeling process seems very analogous to an interpreted programming language versus a compiled one. So, instead of constructing a model and executing it to see if it works properly, construction and execution are performed in an interactive fashion. Chapter 8 describes SL, an implementation of this alternative, model-development environment. As in the previous chapter, an extensive example illustrates the technique and its workings. Chapter 9 contains a summary of some major ideas presented throughout the book, and future directions and conclusions of this research round out the text.

A plethora of pseudo-code algorithms fill the bulk of this text. Alone, that is not bad. However, the authors could have better prepared the reader for these algorithmic excursions by using a few more graphic illustrations to show how the different system components and algorithms interact. Chapter 7 includes just such a figure; I found myself referring to it repeatedly to understand the EL system. The authors interjected many helpful examples, often the same example repeatedly, to illustrate their methods. These examples help to combat the paucity of explanatory figures.

My other criticism of the book relates to Chapters 6 and 7. The simulation modeling environment, EL, presented in these two chapters seems like a confusing digression from the thread that runs through the other chapters. The axiomatic approach of Chapter 4 appears again in Chapter 8. Its use in this later chapter seems like a natural transition from its introduction in Chapter 4. The interjection of the problem description and program-code schemata topics of Chapters 6 and 7 proved to be a disconcerting departure for me as a reader. I suspect that the organization of topics in the book closely follows their research efforts over this time, and therefore Chapters 6 and 7 represent an avenue of research that was later abandoned in favor of an extension to the approach in Chapter 4. Although these two chapters are not without their merit, they produced a logical segue that transitioned poorly between their chapters on an order–sorted logic axiomatic system.

The main contribution of this book is to introduce the concept of logic-based programming to the design and implementation of simulation models. The authors do a good job of describing and illustrating the advantages of this approach, which are modularity, explicit model structure, and model flexibility. Many of the ideas that they propose for the activity of modeling in general are well conceived and are strongly supported.